Implementation of the ANSI T1M1.5 GNM-T1.214 within an ODBMS Framework*

Salvatore Torrente Salvatore D. Morgera K. Wayne Lester

Information Networks and Systems Laboratory
McGill University, Dept. of Elect. Eng., Room 811
3480 University Street, Montréal, Québec, Canada H3A-2A7

Operational Systems Planning
Bell Canada, 160 Elgin Street
Ottawa, Ontario, Canada K1P-6H5

Abstract

Results and observations gathered while implementing the ANSI T1M1.5 GNM-T1.214 standard within an Object-Oriented Database Management System (ODBMS) framework are discussed. The Generic Network Model (GNM) proposes a dynamic network modeling methodology which encompasses a wide spectrum of managed telecommunication network resources. Within the GNM, managed telecommunication resources are modeled from building blocks of lower level resource elements (logical or physical). The complex relationships between GNM resource elements are described either by specialization/generalization or user/containment; therefore, an ODBMS is well suited for implementation of the GNM as compared to conventional database management systems. The ODBMS framework described also allows the database model to evolve with the GNM. Finally, it is demonstrated that a proposed refinement of the GNM into subclasses results in a functional database design that can model a large variety of network resources in accord with a standardized network view.

1 Introduction

The *Advanced Intelligent Network* (AIN) is a catchphrase for the manner in which the public telecommunications network will evolve in the future[1]. There is general agreement that the AIN will capitalize on the growing synergy between computing and communications so as to distribute functionality within the network in a manner such that optimum delay and throughput performance are achieved *independent* of new service creation and delivery. Realization of the AIN requires a *service independent methodology* for distribution of functionality and resource provisioning. A key step to meeting the engineering challenges posed by the AIN is to understand the role that network information models and database structures[2] play in evolutionary architectures. This work describes our experience with a generic network model standard implemented within an object-oriented database framework. The generic network model that is implemented is a functional prototype that we plan to use in an ongoing study on the flexibility and performance of new generation database management schemes built using the object-oriented paradigm[3]. We developed the prototype to assess database management performance for both local and wide area networks. The long term study also examines transaction management techniques, data migration schemes for distributed database systems, and fault recovery systems for distributed data.

2 ANSI generic network model

New telecommunications services are offered to customers every year and often these services require higher bandwidth and greater network integration. In most cases, advanced communication services are built up by combining services from several telecommunication resource suppliers. Each supplier is proprietor of its equipment and sets the rate for its services and equipment use. In the long run, such resource diversity can lead to chaos in resource management unless a *standard network model* is used which clearly defines the *managed* resource elements comprising a customer service across resource suppliers. The T1M1.5 Working Group of the Accredited Standards Committee on Telecommunications - T1, of the American National Standards Institute (ANSI), has proposed such a Generic Network Model (GNM) for Opera-

*Partial support provided under a research contract on *Open Distributed Systems*, Bell Canada, Ottawa, Ontario, Canada, K.W. Lester, Scientific Authority. Partial support also provided by the Canadian Institute for Telecommunications Research, a Government of Canada Network of Centres of Excellence.

tions, Administration, Maintenance and Provisioning (OAM&P) , and, in particular, for interfaces between operations systems and network elements. This work uses the ANSI T1M1.5 GNM-T1.214 as the starting point.

2.1 The role of ANSI in GNM

Current telecommunications networks are populated by a large and increasing volume of Operations Systems (OS) and Network Elements (NE) supplied by different vendors. Both the number and variety of networks and services have grown, creating a diversity of network management needs. This growth has resulted in the proliferation of unique communication interfaces between OSs and NEs. The telecommunications industry stands to benefit from the standardization of these interfaces, designed to achieve interoperability between a broad range of OSs and NEs. The GNM as proposed by ANSI T1.214 is part of a series of standards that specify interface requirements for OSs and NEs. It describes a generic network model needed to develop OAM&P application message standards for modern telecommunications networks. The term *generic* implies that managed object classes and their properties as described are applicable across different telecommunications technologies (e.g., SONET, ISDN) for various OAM&P functions defined in ANSI T1.210. The GNM T1.214 standard is a framework for defining management information to be exchanged by implementations of the Common Management Information Service Element (CMISE)[4].

2.2 Overview of ANSI GNM

The GNM proposed by the T1M1.5 working group of ANSI is a dynamic resource provisioning model, developed to encompass a large spectrum of communications resources. In particular, the GNM offers a set of resource models which can be composed of other resource models as required, both at the physical and logical levels. The physical level of the model represents all of the core hardware used to implement a service or network of services. The logical level of the GNM is composed of resource models constructed from lower level core resource models (physical or logical), down to the basic hardware elements composing the equipment. The resource models are distributed across the network. Each resource model may be composed of equipment from different suppliers and, within the GNM, are viewed as *managed object classes* of telecommunication resources. The Generic Network Model is essential to the generation of uniform fault, configuration, performance, security, and accounting management standards. A common network model, identifying the generic resources that exist in a network and their associated attributed types, events, actions, and behavior, provides a foundation for understanding the interrelationships between these resources and attributes, and, in turn, promotes uniformity in dealing with the mangement of these resources. Resources have attributes that allow the user to control and observe the behavior of the resource. Attributes allow the user to monitor and control the relationships between resources. A complete documentation of managed object classes for the GNM can be found within the ANSI standards document T1.214 from ANSI[4].

2.3 GNM - An experimental model

The GNM is a complex resource modeling framework which encompasses a very large variety of telecommunication resources. For our prototype, we extracted a subset of the original GNM and focused on the basic telecommunications resources that may comprise a network service. The basic elements of concern in our design are carriers (line, channels), node terminating equipment (MUX's, circuit packs, ports), and carrier routing equipment (digital cross-connection systems, carrier cross-connections). These resource models are sufficient at this time to develop a database design supporting the construction and maintenance of basic network services. In the GNM, the resource models are viewed as managed object classes. In the prototype, the following managed object classes are used:

Network: A collection of interconnected telecommunications and managed objects (logical or physical) capable of exchanging information.

Equipment: Telecommunications equipment comprising a network element.

Network Element: Either groups or parts of telecommunications equipment, where the component equipment might be geographically distributed.

Line: Physical transmission medium.

Line Termination: The end connection of a line.

Channel: The sections of end-to-end information path specific to one line.

Channel Termination: The ends of a channel.

Cross-Connection: An assignment between two channel terminations.

Path: A connection characterized by a specified rate which is independent of the physical means of carrying the signal.

Information Path: Subclass of path, which carries information from point-to-point.

Path Group: Parallel paths of the same type.

3 The GNM within an ODBMS framework

Our research centers around the implementation of the GNM within a database system. Since the GNM proposed by the T1M1.5 Working Group views telecommunication resources as classes of *managed objects*, an object-oriented approach to the problem is appropriate[3]. In particular, within the GNM, managed objects (resource elements) share many common attributes, either in behavior or in the data they hold.

3.1 Overview of ODBMS

The constructs and vocabulary of object-oriented databases are based on object-oriented programming languages. At the root of object-oriented database models are the *objects* themselves[5]. Objects are entities that encapsulate data with procedures (or methods) that manipulate the data and communicate by means of messages. For the prototype, an ODBMS offers several advantages over conventional database systems that are worth mentioning. First, an ODBMS offers a more realistic data model; classes and objects in an ODBMS represent real-world design entities and give a much better feel for the mechanics of the problem. Second, an ODBMS offers a more powerful data model; object data representations are highly flexible and can be easily customized with little restriction. Third, with the GNM's complex class-object relations between resource models, an ODBMS offers an easier schema development; generalization and inheritance allow the schema to be better structured, more intuitive, and capture the semantics of the application[6]. The ODBMS offers a powerful unified knowledge representation scheme. The object-oriented representation of information provides a uniform framework for integrating data (attributes) and knowledge (procedures). Finally, since the GNM can contain several resource models with identical behavior and state, the ODBMS object identity system can relate two or more objects independently of their embedded values. This latter point implies that identical managed objects within a GNM class can share attributes.

3.1.1 A proposed ODBMS environment for the GNM

An important aspect in the prototype development was the selection of an ODBMS environment suitable for the implementation of the GNM[3]. Our implementation of the GNM is also a study of database management techniques for distributed data. In particular, we have already indicated that the GNM is a resource modeling framework; this framework may extend over a wide area covering hundreds of resource elements, all interdependent, yet owned by different suppliers. A wide area distribution of network services would require several databases sharing information, each database responsible for maintaining the data integrity of its local area network, while the combination of data from all databases would reflect the entire GNM for the wide area distribution of network customer services, as shown in Figure 1.

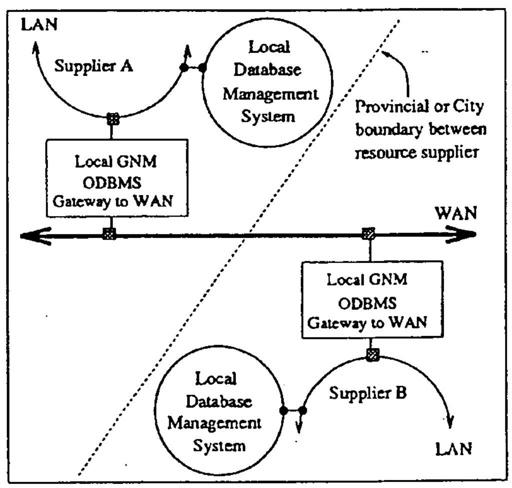

Figure 1: Distributed GNM over a WAN

For each database to be a node within the entire distributed system, the database must offer a portion of its storage area for information sharing (protected disk partition) to a selected group of databases over a wide area network, while maintaining a private area (private disk partition) for local services and information updates. The private disk partition of a database contains the local services of a resource supplier, while the shared disk partition contains all information on services available from the local supplier for wide area network customer services. Since data within the GNM will be shared amongst several database man-

agement systems, the ODBMS must support *concurrency control* for multiple access to the shared data partition within a node, as depicted in Figure 2.

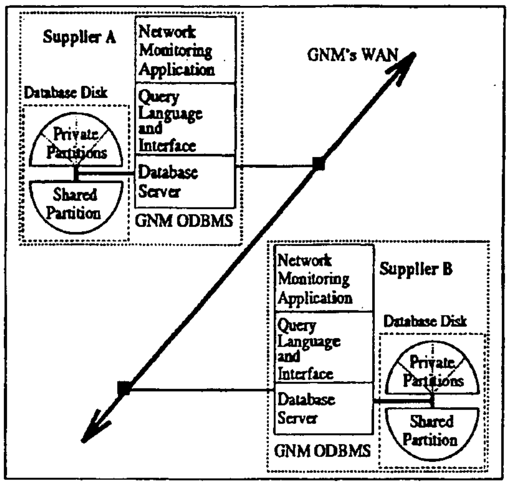

Figure 2: GNM database design for resource suppliers

For adequate system development, the ODBMS environment must support high level database design tools to assist the implementation of the GNM prototype and its maintenance within a distributed network. The database system must offer an expressive query language for complex data retrieval and manipulation. The query language must be accessible through a programming interface with support for alarms and transaction notification to the user interface. Development and maintenance of higher level software interface (GUI or other) must be independent of the database's own programming code upgrades. The internal programming language of the database must support a complete encapsulation of the GNM's dynamic structure within a manageable set of methods. Due to the distributed nature of the GNM, which is meant to transcend the boundaries of local telecommunications resource suppliers, the ODBMS environment must be designed for a distributed network of *heterogeneous* database systems. The distributed network supporting the GNM may require tapping into a local suppliers conventional database system for resource monitoring; hence, the selected ODBMS environment must address the issue of standard query languages. Current ODBMS's lack a high-level standard query language. Various object-oriented query formalisms are currently under development[7]; in particular, ObjectiveSQL (OSQL) is an object-oriented approach to SQL. ISO and ANSI are moving toward a standard for the Structured Query Language (SQL). They are also moving toward a standard technique to pass information between clients and servers in a database environment, using Remote Data Access (RDA)[8]. This will permit interoperability between ODBMS's and existing data in dissimilar database systems, as shown in Figure 3. The SQL Access Group was formed to accelerate this standardization process.

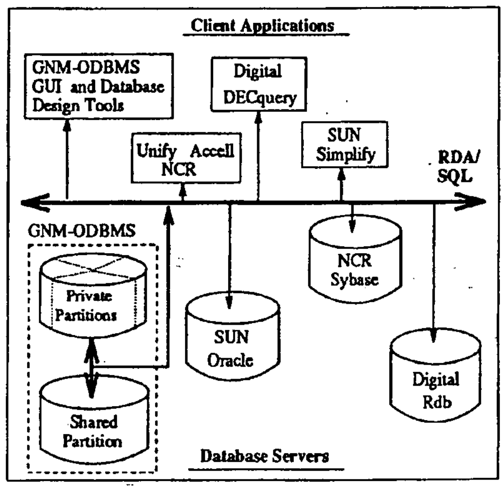

Figure 3: GNM on a heterogeneous database network

3.1.2 Itasca ODBMS

The database selected for this work is the ITASCA[1] object-oriented database system from Itasca Systems Inc. The ITASCA database is an active object database management system. The system stores and activates methods directly in the database; hence, the database design is language neutral[8]. ITASCA is based on a series of object database research prototypes, initiated by the Microelectronics and Computer Technology Corporation (MCC) Object-Oriented and Distributed Systems Laboratory under the ORION project. The ITASCA product is an extension of the ORION-2 prototype from MCC, a multi-client/multi-server architecture, with added features and improved functionality over its ORION ancestors. ITASCA employs a distributed architecture with shared objects spread across UNIX[2]-based computers on a local area

[1]ITASCA is a trademark of Itasca Systems, Inc.
[2]UNIX is a trademark of AT&T

network. The ITASCA model follows the object-oriented view that uniformly models any real-world entity as an object. Each object has a unique identifier along with a state and behavior. A class object collects objects that share the same set of attributes and methods. Subclasses derive from existing classes. The resulting database definition is a class hierarchy, with support for multiple inheritance. ITASCA supports database management features from both conventional database systems and object-oriented database systems, in particular, ITASCA offers long-duration transactions, distributed transaction management, object migration, and shared and private databases as shown in Figure 4.

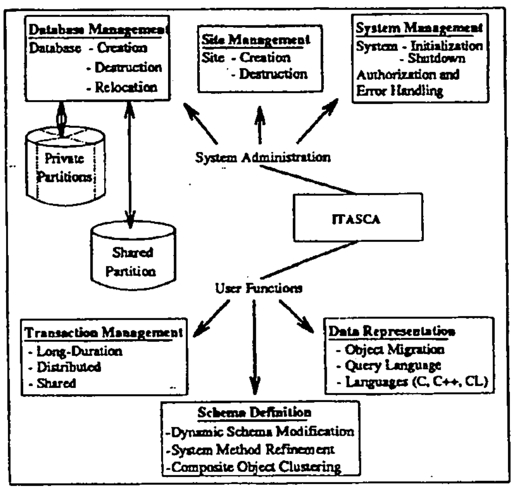

Figure 4: ITASCA functional overview[3]

The shared and private databases exist in a distributed environment in ITASCA; this makes the product very appealing for experiments with the GNM, since it is possible to create the GNM *within* the shared database, while leaving the private database as a repository for private resource management without affecting GNM design or performance. An ITASCA server controls the partition of the shared database at each site. ITASCA clients are provided transparent access to the various partitions of the shared database. ITASCA allows any number of private and shared databases. Private databases allow private data that is not shared with other users of the database. Long duration transactions allow users to check objects out of the shared, distributed database into their private databases as shown in Figure 5.

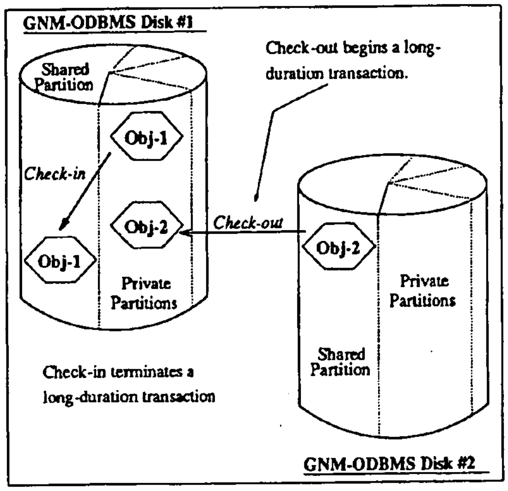

Figure 5: Long duration transaction[3]

Users can then change the objects in the private databases without affecting the shared database or other users. These changes can be committed to the private database. Then, at any later time, the user can check the updated object or objects back into the shared database. Change notification in ITASCA is either flag-based or message-based. Flag-based notification will identify an updated object upon querying the object for such information. It is a passive notification scheme. Message-based notification, on the other hand, is an active notification scheme. It will execute a method (or code) upon an update or other change to an object. Such methods can send mail messages or invoke other methods or programs. Within ITASCA, management of schema updates is automatic for all sites, including sites that were off-line during any changes. ITASCA stores each instance of data in one site. The system or a user may move the data from one site to another to improve data locality. Within the ITASCA architecture, access to moved data remains transparent, and there is no need for a user/application to know the specific location of the data within the distributed database. ITASCA supports a Schema Editor GUI and Browser for the development of the GNM within the ODBMS. ITASCA also supports a Database Administration Tool GUI (DBA tool) for database disk partitioning and disk access authorization to other databases on the network.

[3]Adapted from *Distributed Object Management Systems*[8]

3.2 A proposed database definition for the GNM

From the GNM, we can define several class definitions for telecommunications resources within a network service. We first describe the class definition for a set of core classes from the GNM; these core classes are the building blocks of the database definition for the GNM. From the core classes, other classes are developed and described in order to properly implement the GNM resource model definitions. Taking a hierarchical approach to the problem definition, the highest level of the hierarchy contains the network service[9]. Within the network service, all other components (physical or logical) are subclasses or siblings of the network class. As described earlier, the managed objects within the GNM can be composed of other managed objects contained within the GNM. Network services can be composed of other networks, constructed of more complex or simpler models; hence, the database definition of the GNM should contain a class network with a class network attribute, as shown in Figure 6. The arrows indicate how the *top* network representing the network service is built from inheriting the data and functionality of several subnetworks of class *Network*.

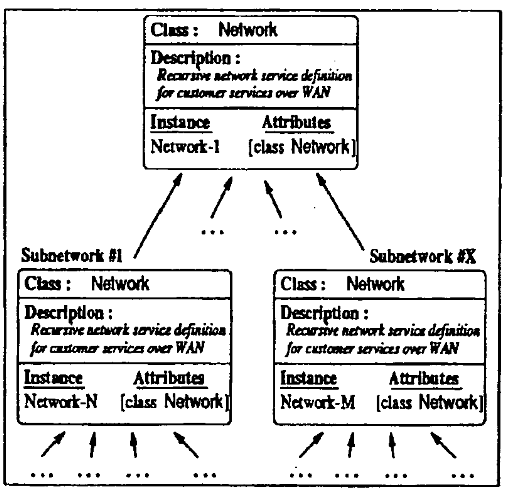

Figure 6: Example definition of class *network*

Within the GNM network, elements are the building blocks of a network service. Networks are composed of Network Elements (NE), elements that are considered equipment, such as multiplexers (MUX), circuit packs, and digital cross-connect systems (DCS), as in Figure 7.

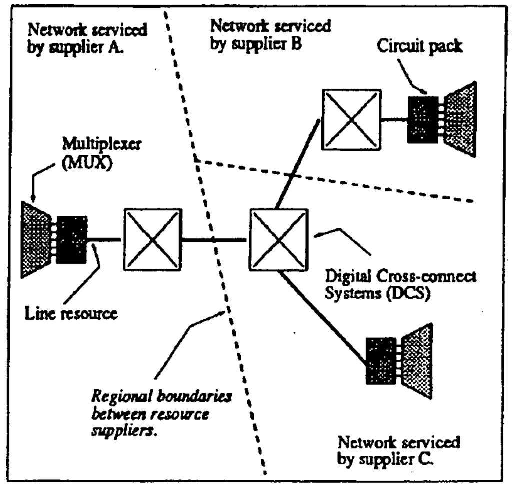

Figure 7: Example network service model

A line is also considered a managed object. The *Line* class is a *carrier* resource element within the GNM and cannot contain other lines, but channels can be derived from a line. A *Channel* object is said to be in a *containment* relation with the *Line*. In addition, channels can be derived from higher rate channels by splicing channel rates into subchannels. The *Line* class uses the *Network Element* class to terminate its connections from point-to-point at circuit packs. A *Circuit Pack* is a network element in a special *user* relation with the other network elements such as MUX's and DCS's. The *Circuit Pack* subclass of the parent class *Network Element* is used by all other network element subclasses as a linking port to the line carrying the network service. In accord with this discussion, the core database definition for the GNM implementation within an ODBMS starts as shown in Figure 8.

3.2.1 Mapping the GNM within an object-oriented database model

In the previous section, we took a simplifying approach to the GNM and identified the basic core classes required to implement the GNM within an ODBMS. Several other classes are required to meet the GNM definition, and we now define the remaining classes. We have shown that network elements are network service equipment; equipment can be either

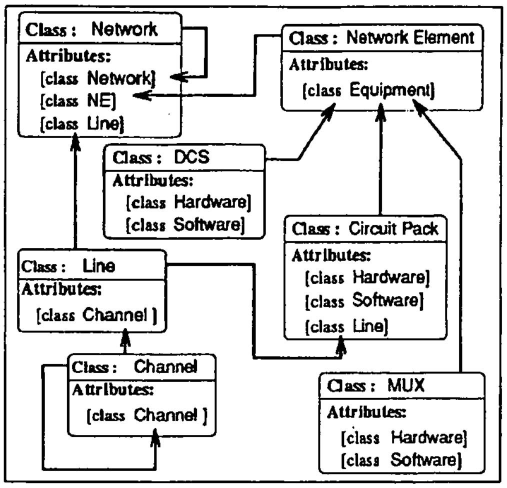

Figure 8: Core GNM class definitions and relations

hardware or software, or a combination of both. The GNM is a mapping of heterogeneous hardware and software into a standard network model. The network model is composed of elements from two sets, one set is the physical elements within the GNM and the other set is the logical elements within the GNM. Although the logical set is built from basic hardware and software elements present within the physical set, the distinction between the physical and logical set of resource elements is an important one. In particular, the GNM is intended to simplify the *operation and management* of network telecommunication resources; hence, the elements composing the network must be managed at a high level of abstraction, where the intricate details of every physical resource composing the network elements are encapsulated within a functional building block. We now present the details of the managed objects within both the physical and logical sets of network elements. The managed objects of interest in our GNM implementation are lines, circuit packs, multiplexers, digital cross-connections, and carrier cross-connects, for the physical set, and channels, information paths, path groups, and networks, for the logical set. We also discuss the relationships between the elements of each set.

3.2.2 Database model for the GNM's physical elements

Lines are the physical elements supplying network services within the model. Although a line itself may be constructed from several stretches of conducting medium for a given line rate, our prototype considers the *Line* class object as a base class and building block within the physical set. Lines require a line termination point to the circuit pack in order to properly match the GNM definition. These line terminations are ports to the *Circuit Pack* class. Each network element node can only interconnect two or more lines through the line termination ports to the circuit packs. Since the class *Line* is a base class within the physical element set of the GNM, *Line* objects influence the behavior and state of all classes inheriting their resources. The main attributes found for line objects are line rate, operational state, circuit pack port at point A, and circuit pack port at point Z.

As stated earlier, the circuit pack is an *equipment* subclass with a special user relation to other network elements. Even though the circuit pack is tightly coupled in behavior and operational status with its assigned node terminating equipment, the GNM permits the circuit pack and node terminating equipment to be geographically dispersed. The circuit pack building block is a composite object of hardware and software resources. In addition, since node terminating equipment such as MUX's, DCS's, and carrier cross-connects are composed of basic hardware and software elements, the class *Equipment* is viewed, at the physical level, as a generalization of class *Circuit Pack* and of classes *Hardware* and *Software*, composing network elements, as depicted in Figure 9.

The class *Hardware* and class *Software* have attributes which reflect their operational state, as reported by sensing devices and report stations, from individual equipment components used by the network.

3.2.3 Database model for the GNM's logical network

The logical set of resource elements contains the most complex and intricately related models of the GNM. At the logical level of the GNM, network building blocks encompass physical elements to create active class objects that inherit behavior and information from parent and child classes, thereby implementing the dynamics of the GNM. Above the class *Equipment*, within the physical set of network models, lies the class *Network Element*. The class *Network Element* is a logical class representation of all equipment building blocks which inherit from the physical *Equipment* class. In our work, the physical set of terminating node equipment is mapped within the logical set of network elements, as shown by Figure 10.

The class *Network Element* inherits only the build-

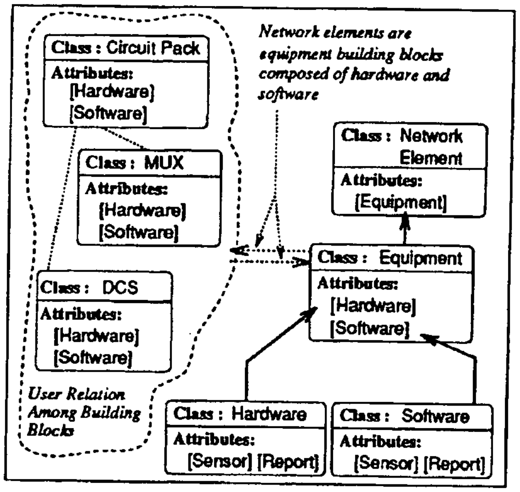

Figure 9: Model for physical set of network elements in the GNM

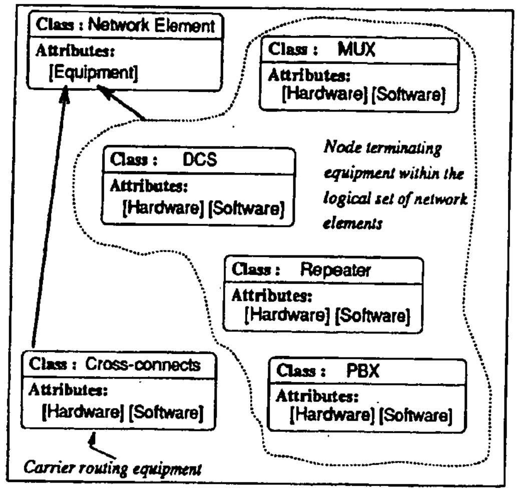

Figure 10: Mapping equipment to the logical set of resource components

ing blocks of equipment composed of hardware and software (MUX's, DCS's, ...), and not the specific list of hardware and software present within the building blocks. Network service monitoring and management benefits from this data abstraction, by eliminating much of the underlying lists of equipment reports. A building block within the GNM is functional if all underlying hardware and software (class *Equipment* in the physical set) are functional. The details of any malfunction within the hardware or software can be queried from the database at the class *Equipment* level only when necessary. The network manager, therefore, need only monitor the building blocks (network elements) of the model to assess the model's functionality, while technical staff can directly query the class *Equipment* for diagnostics on key equipment support.

As described in the previous section, the class *line* resides within the physical set of our GNM implementation. Although a line within a network is the carrier of services, a service does not necessarily use the full bandwidth of a line. At the logical level, our implementation breaks down the physical line into a set of N channels, where the combined bandwidth of all channels derived from a line does not exceed the total line bandwidth. Here again, we have a mapping of resources from the physical set of resource elements to a logical set. A *Channel* is a logical representation and a base class in our GNM implementation, from which all services will be carried to a user. Since a channel cannot exist without a line (although a line does exist without channel assignment), the class *Channel* is contained within the class *Line*, and the class *Channel* inherits, by *specialization*, the behavior and status of its parent class *Line*.

Network services are supplied to a customer in groups of channel assignments. These channel assignments are considered a *Path Group* in the GNM. A *Path Group* object is composed of several paths, themselves built from a sequence of end-to-end channels riding on line segments between two points of service.

A *Path* object is created by assigning an end-to-end sequence of channels. In the prototype, the channels selected to create a path have all the same rate; hence, the class *Path* is a generalization of class *Channel*, where the *Path* objects are lists of channels assigned to a particular information path offered to the customer. Since the service available from a path is limited to the bandwidth of its composing channels, a network service is built from a group of paths, making the class *Path Group* a generalization and parent class of the subclass *Path*.

Figure 11 presents the logical level structure of the GNM as implemented within the ODBMS. The GNM presented in Figure 11 focuses on the core classes implemented within the ODBMS, with several other classes such as *Framed Path, Connection On Box, Spans, Span Terminations*, etc are also implemented in our GNM prototype. A complete description of the managed object classes can be found in the ANSI

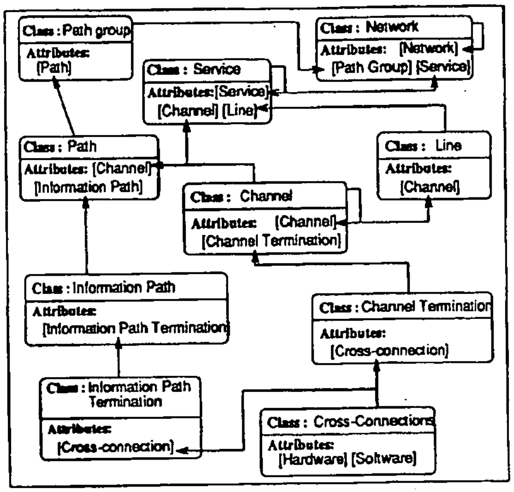

Figure 11: GNM prototype implementation

T1.214 standard[4]. The model in Figure 11 is composed of core classes and links between classes representing their relation within the GNM. The directional arrows indicate the sense of model construction, where the tails of the arrows are the managed object classes which compose the targeted parent class structures. Class objects with self-pointing arrows highlight the class behavior of recursive class definition: objects composed of other identical *class* objects. Each class within this logical representation inherits attributes and methods from a parent or child class, either at the logical level or at the physical level. The class *Line* contains a method for attaching the line extremities to the termination ports of the circuit packs. In addition, the class *Line* contains a method for deriving channels from a line; this method is inherited by the subclass *Channel* also, which uses it to derive subchannels from each parent channel. Channels of a line and subchannels can be derived until all available bandwidth on the carriers are exhausted, with no other restriction imposed upon the recursive channel assignment. The channel functionality is inherited from the parent line.

Figure 11 shows an additional class *Service*. This class has been added to conform with the proper T1.214 class definition for network resource assignments within the GNM. The resources inherited by the class *Service* are all line and channel resources made available for network services. At the managerial level, a query requesting all services available for network support would be made to the class *Service* that would list all network service resources found within the public partitions of the distributed database system. The class *Path Group* contains a method for channel grouping; this method requires the use of class method within the subclass *Path* for channel identification. This *hierarchical method calling* greatly simplifies the development of the GNM's functionality. From a set of core methods, instance and class methods can be developed for all the logical level design of the GNM; this has the advantage that any modification to a class or instance method will be distributed to all derived methods. The ODBMS design presented in Figure 11 was sufficient to implement a base prototype of the GNM-T1.214 for experimentation.

4 Conclusions and Recommendations

Our experimental implementation of the GNM spanned approximately one year of analysis and design and is ongoing. The object-oriented implementation of the GNM within the ODBMS followed the sound and proven Object-Oriented Analysis (OOA) and Object-Oriented Design (OOD) techniques as presented by Booch[5] and Kim[9]. We believe our database definition of the T1.214 standard accurately implements the dynamics of the GNM as proposed by T1M1.5 working group at this time.

Implementation within an object-oriented database framework proved to be invaluable for the development of the GNM prototype. The ODBMS framework substantially eased the process of incremental database design and prototyping and testing by permitting a dynamic redefinition of the GNM prototype while the database was up and operational on LAN. In particular, ITASCA's support for shared and private database partitions greatly simplified the procedures required in fine tuning the GNM prototype. Entire data structures of the GNM prototype could be extracted from the shared database partition to the private disk for redefinition, without corrupting or disabling the remaining GNM model prototype.

Figure 12 presents a sample session of the GNM model through our prototype GUI program communicating with the database. The GUI displays physical transmission lines on a geographical map, as entered by the users, while *managed telecommunication resources* such as customer circuits are displayed in a *Path Group Display* window for further analysis and monitoring. All the methods necessary to implement the GNM's dynamic behavior were constructed with minimal coding and were accessible at all times within

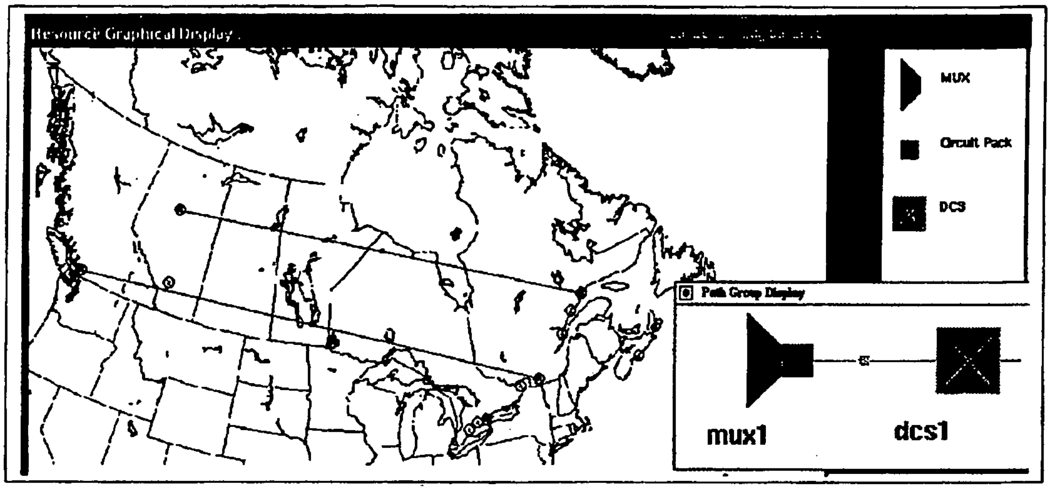

Figure 12: Example GNM session using a protoype GUI to the ODBMS

the database definition of the GNM. The ODBMS offered, by far, the best implementation environment for the development of the GNM prototype.

The ODBMS performance during our experiments indicates that the database seek time, for data retrieval, is constant with respect to the number of classes defined within the database. On the other hand, database seek time increased nonlinearly with the number of instances per class. As the number of class instances increased linearly among several classes the time delay to extract data increased by orders of magnitude and the data was delivered at the interface at an irregular rate. Although some of the delays observed from the database can be of legitimate concern in some cases; we believe that these concerns are minor when viewed in terms of the versatility and development power offered by the ODBMS environment.

References

[1] R.B. Robrock II, *The intelligent network-Changing the face of telecommunications*, **Proc. IEEE**, Vol. 79, pp. 7-20, Jan. 1991.

[2] See **IEEE Journ. Sel. Areas Commun.**, Special Issue on Databases in Communications Systems, Vol. 7, April 1989.

[3] B. Ellacott, S. Torrente and S. D. Morgera *The Object-Oriented Paradigm in a Distributed Network Management Environment*, Research Report, McGill University, October 1991.

[4] Exchange Carriers Standards Association, *Operations, Administration, Maintenance and Provisioning (OAM&P) - Generic Network Model for Interfaces between Operations Systems and Network Elements*, American National Standards Institute, Inc., T1.214-1990.

[5] G. Booch, *Object-Oriented Design, with applications*, The Benjamin/Cummings Publishing Company, Inc, 1991.

[6] P. Coad and E. Yourdon. *Object-Oriented Analysis*, Yourdon Press, Englewood Cliffs, NJ, 1990.

[7] G.M. Shaw and S.B. Zdonik, *Object-oriented queries: Equivalence and optimization*, Proceedings of the 1st International Conference on Deductive and Object-Oriented Databases, December 1989.

[8] Itasca Systems Inc., *ITASCA, Distributed Object Management System, Release 2.0*, Minneapolis, Minnesota, 1991.

[9] W. Kim, *Introduction to Object-Oriented Databases*, The MIT Press Cambridge, Massachusetts, 1990.